NYU Tandon School of Engineering
Computer Science and Engineering
CS-GY 6083 sections A and INET, Fall 2024
Prof Phyllis Frankl

Project Part 1

A non-profit organization **WelcomeHome** similar to Ruth's Refuge needs a better way to track donations, orders, deliveries, clients, donors, and volunteers. WelcomeHome's clients are refugees and asylum seekers who have found housing, but need housewares and furniture. Donors donate used housewares (dishes, cookware, small appliances, decorative items, etc), used furniture, or money to buy new items. WelcomeHome stores these items in one of its storage facilities. Clients use a portal similar to an online shopping service to create orders indicating which of the available items they need/want. After a client has ordered, volunteers collect and label the physical goods and (maybe a few days later) other volunteers and/or staff members deliver the order to the client's new home.

In Part 1 of the project, you will design an ER model for WelcomeHome's database, capturing the organization's requirements, detailed below. In Part 2 of the project, the Professor and TAs will provide our solution to Part 1 and you will use it to design a relational database schema and write some queries. In Part 3, you will design, implement and test application code for a web-based application WelcomeHome's staff and volunteers can use to track the location of items, track collection of orders, track order status, etc.

ER requirements:
The WelcomeHome database will store data about People (staff, volunteers, clients, donors), Items in the inventory, storage locations within the warehouse, and Orders that are in progress. Details follow.

[**Hint:** terms in **bold** should be modeled by (weak or strong) entity sets or relationship sets.]

There are various kinds of people with various *roles* (e.g. staff, volunteer, client, donor). Each **Person** has a unique username**;** additional information that may

be stored about a Person includes their name, composed of first name, middle name, last name, email address, and zero or more phone numbers, their primary language, zero or more additional languages they speak, and a password. If you'd like, you may use specialization/generalization to distinguish between various roles different People play, but it's OK if you lump them all together and include a *role* attribute to make this distinction.

Each donated or purchased **Item** has a unique itemID, a description, a photo, colors, material, an indication of whether it is new or used, and other notes about the item. Items are associated with a predetermined set of **Categories**, along with sub-categories (e.g. category: cookware; subcategory: frying pan). We'd also like to indicate whether or not an item requires assembly.

Some items have multiple pieces, which may end up stored separately, so we want to also track **Piece**s. (For example, if we had a dining room set consisting of a table and four chairs, we would have five pieces of this item.) Each Piece has a pieceNum which is unique for its item, but different items could have the same pieceNum. Each piece may have a description, a quantity, a length, a width, and a height.

The storage facility has numerous **Location**s where (pieces of) items are **Stored**. The storage facility has many rooms, each with many shelves and/or areas of the floor. A Location is identified by a unique pair consisting of a room number and a shelf number. There may also be a shelf description.

With help of staff or volunteers, clients will browse the inventory and select items that they want to include in an **Order**. This is mainly done via an outside system like Shopify, and we will not be concerned with that aspect in this project (except possibly for importing and exporting bulk data between our system and the shopping system). Rather, the course project will focus on recording the locations of (pieces of) donated items, in order to help volunteers collect orders, and tracking the status of orders and who is moving what where.

Each **Order** will have a unique order ID, a start date, a delivery date, and notes. Each Item can be in at most one order.

There are also various relations between Persons, Orders, Items, etc. We'd like to track **which donor donated which Item**, **which client made each order**, **which volunteers or staff delivered each order,** and **which individual staff member supervised each order**. At this point, you are not required to consider the different roles of people and who does what (i.e. staff supervises, donor donates); Later, we may require that the application code enforces such constraints.

ADDITIONAL REQUIREMENTS:

Draw your ER diagram neatly, either by hand or using a drawing tool. You **must** use the notations studied in class. You may use either the double line and arrow notation for participation and cardinality constraints or the lower-bound .. upper-bound notation. Your diagram should fit easily onto a single page – if you have more than about 15 components (strong or weak entity sets or relationship sets) you should rethink your design.

In the above description, at times the same word is used to describe different attributes (e.g. date, notes, etc … may be ambiguous.) Give different names to different attributes, e.g. "itemNotes" and "orderNotes", instead using "notes" in two different places.

On a separate page, indicate any assumptions you are making about any ambiguous requirements. Also indicate briefly what each (strong or weak) entity set and each relationship set in your design corresponds to from the textual description, above.

You may work alone or with one or two classmates. You may work alone on this part and then team up with others for the later parts of the project, but once you form a team, you should stay with them for the duration of the project (except in very unusual circumstances, such as illness of a teammate or withdrawal from the class). Consequently, if you are forming a team for part 1, you should think ahead to what programming language(s) you'll use for the application programming part of the project later. Material on how to implement web-based database applications in Python with Flask and MySQL will be presented in class.

If you are working with others, you must notify the TAs at least one week before the due date. A form will be provided for doing so.